CPE/EE 421
Microcomputers

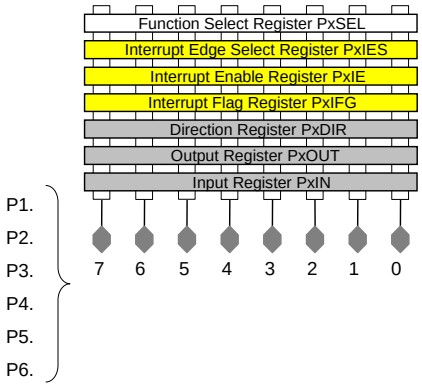
Instructor: Dr Aleksandar Milenkovic
Lecture Note
S17

Course Administration

- Instructor: Aleksandar Milenkovic
  milenka@ece.uah.edu
  www.ece.uah.edu/~milenka
  EB 217-L
  Mon. 5:30 PM – 6:30 PM,
  Wen. 12:30 – 13:30 PM
- URL: http://www.ece.uah.edu/~milenka/cpe421-05F
- TA: Joel Wilder
- Labs: Lab #4 is on. Hw #2 is posted.
- Test I: Graded. Solutions are in scr/.
- Text: Microprocessor Systems Design:
  68000 Hardware, Software, and Interfacing
- Review: M68K (Chapter 1; Chapter 2; Chapter 3),
  MSP430 (Introduction, Arch., Basic Clock System,
  WDT, Low Power Modes, Digital I/O)
- Today: MSP430 Digital I/O, Timers, USART

Review: Digital I/O

all MSP430



Port1 Port2	Port3 ... Port6
yes	yes
yes	no
yes	no
yes	no
yes	yes
yes	yes
yes	yes

Chapter 9, User's Manual
pages 9-1 to 9-7

Digital I/O Introduction

- MSP430 family – up to 6 digital I/O ports implemented, P1-P6
- MSP430F14x – all 6 ports implemented

Ports P1 and P2 have interrupt capability.

Each interrupt for the P1 and P2 I/O lines can be individually enabled and configured to provide an interrupt on a rising edge or falling edge of an input signal.

The digital I/O features include:

- Independently programmable individual I/Os
- Any combination of input or output
- Individually configurable P1 and P2 interrupts
- Independent input and output data registers

The digital I/O is configured with user software

Digital I/O Registers Operation

Input Register PnIN

Each bit in each PnIN register reflects the value of the input signal at the corresponding I/O pin when the pin is configured as I/O function.

Bit = 0: The input is low

Bit = 1: The input is high

Do not write to PxIN. It will result in increased current consumption

Output Registers PnOUT

Each bit in each PnOUT register is the value to be output on the corresponding I/O pin when the pin is configured as I/O function and output direction.

Bit = 0: The output is low

Bit = 1: The output is high

Digital I/O Operation

Direction Registers PnDIR

Bit = 0: The port pin is switched to input direction

Bit = 1: The port pin is switched to output direction

Function Select Registers PnSEL

Port pins are often multiplexed with other peripheral module functions.

Bit = 0: I/O Function is selected for the pin

Bit = 1: Peripheral module function is selected for the pin

Digital I/O Operation

Interrupt Flag Registers P1IFG, P2IFG

(only for P1 and P2)

Bit = 0: No interrupt is pending

Bit = 1: An interrupt is pending

(Only transitions, not static levels, cause interrupts)

Interrupt Edge Select Registers P1IES, P2IES

(only for P1 and P2)

Each PnIES bit selects the interrupt edge for the corresponding I/O pin.

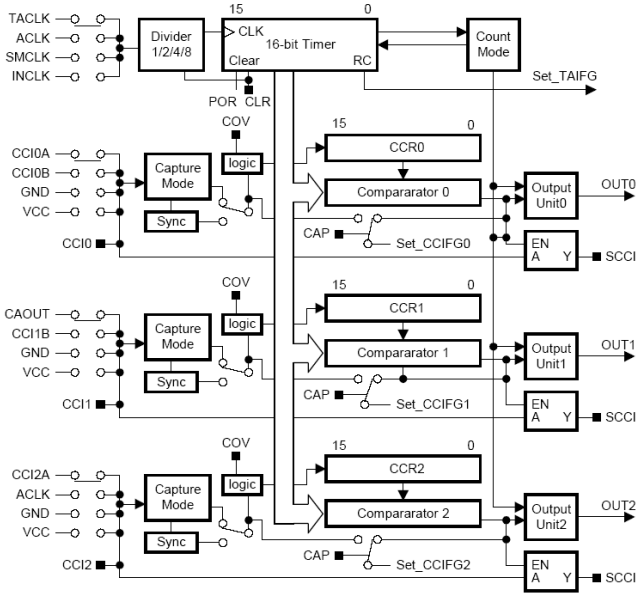
Bit = 0: The PnIFGx flag is set with a low-to-high transition

Bit = 1: The PnIFGx flag is set with a high-to-low transition

Timer_A MSP430x1xx

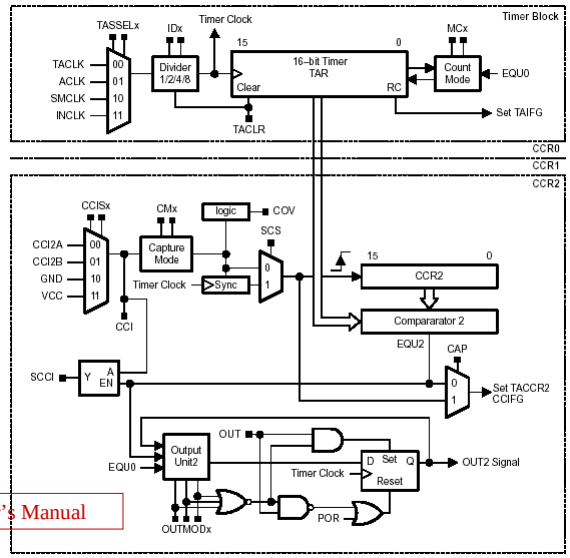
- **Purpose**
  - The Timer A and B systems on the MSP are a versatile means to measure time intervals. The timers can measure the timing on incoming signals or control the timing on outgoing signals. This function is necessary to meet arbitrary timing requirements from outside components, and the ability is useful in phase locking scenarios
- **Features**
  - 16-bit counter with 4 operating modes
  - Selectable and configurable clock sources (internal - ACLK, SMCLK; external – INCLK, TBCLK)
  - Three (or five) independently configurable capture/compare registers with configurable inputs
  - Three (or five) individually configurable output modules with 8 output modes
  - Multiple, simultaneous, timings; multiple capture/compares; multiple output waveforms such as PWM signals; and any combination of these.
  - Interrupt capabilities
    - each capture/compare block individually configurable

Timer_A3



9

Timer_A5 - MSP430x1xx Block Diagram



Page 11-3, User's Manual

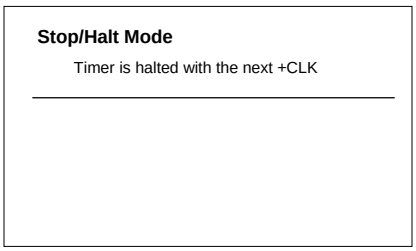
Capture and Compare Registers

- What is a capture?
  - A record of the timer count when a specific event occurs. The capture modules of the timers are tied to external pins of the MSP. When the control registers of timer A and the specific capture compare module have been properly configured, then the capture will record the count in the timer when the pin in question makes a specific transition (either from low to high or any transition). This capturing event can be used to trigger an interrupt so that the data can be processed before the next event. In combination with the rollover interrupt on Capture module 0, you can measure intervals longer than 1 cycle.
- Compare
  - The inverse of a capture. While capture mode is used to measure the time of an incoming pulse width modulation signal (a signal whose information is encoded by the time variation between signal edges), compare mode is used to generate a pulse width modulation (PWM) signal. When the timer reaches the value in a compare register, the module will give an interrupt and change the state of an output according to the other mode bits. By updating the compare register numbers, you change the timing of the signal level transitions.

Timer_A Counting Modes

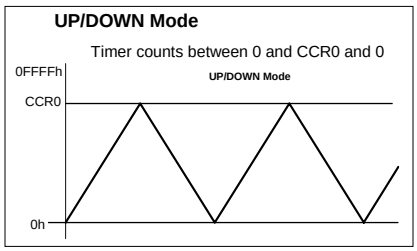
Stop/Halt Mode

Timer is halted with the next +CLK



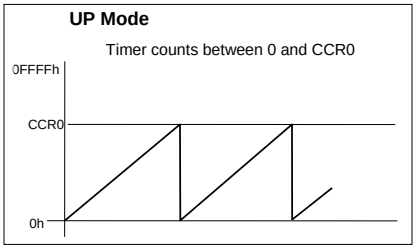
UP/DOWN Mode

Timer counts between 0 and CCR0 and 0



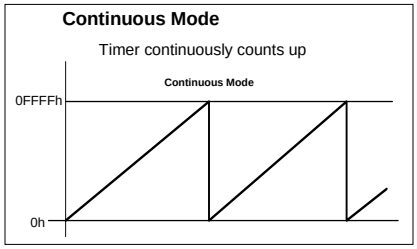
UP Mode

Timer counts between 0 and CCR0

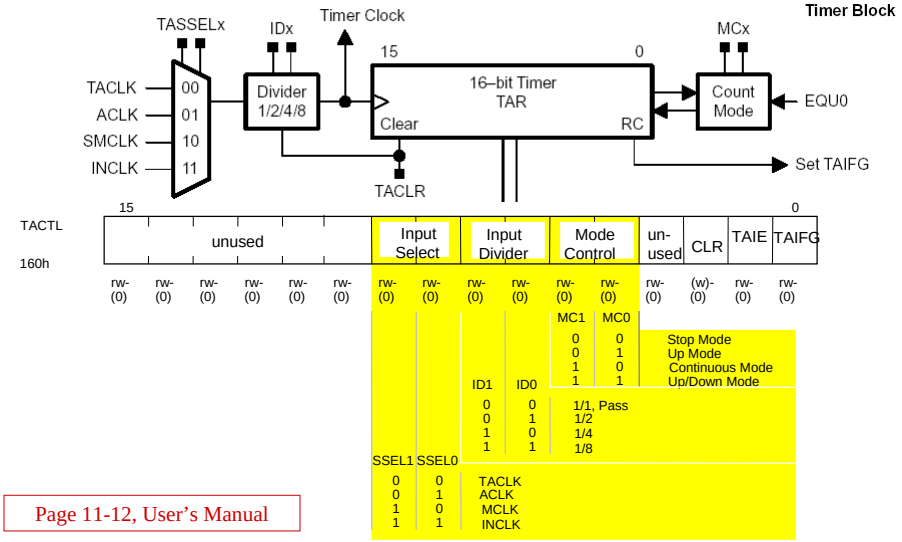


Continuous Mode

Timer continuously counts up

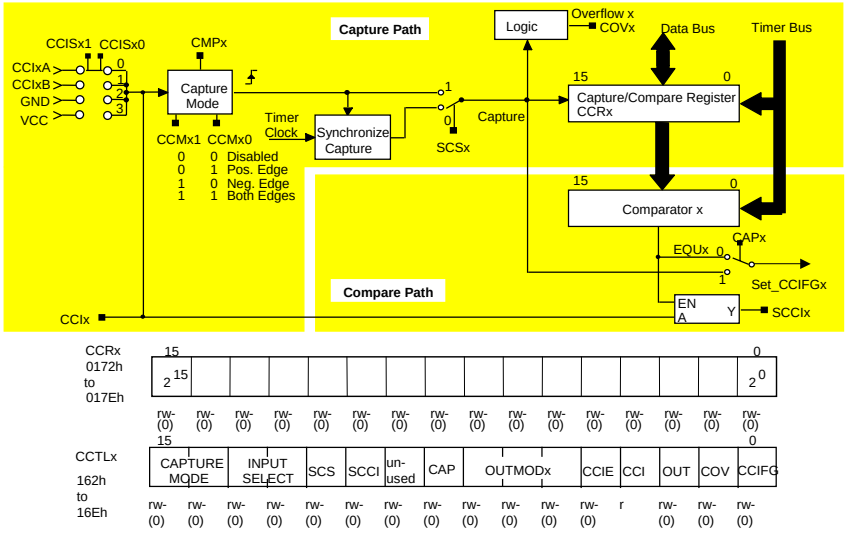


Timer_A 16-bit Counter

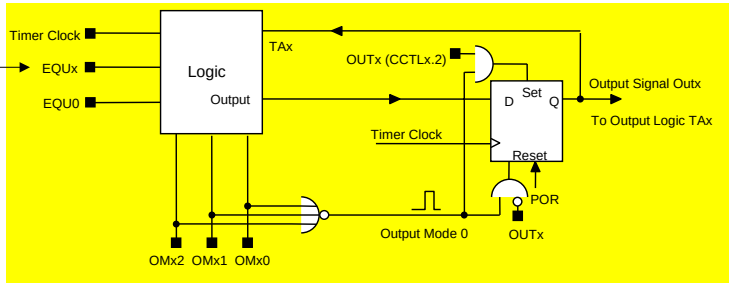


Page 11-12, User's Manual

Timer_A Capture Compare Blocks

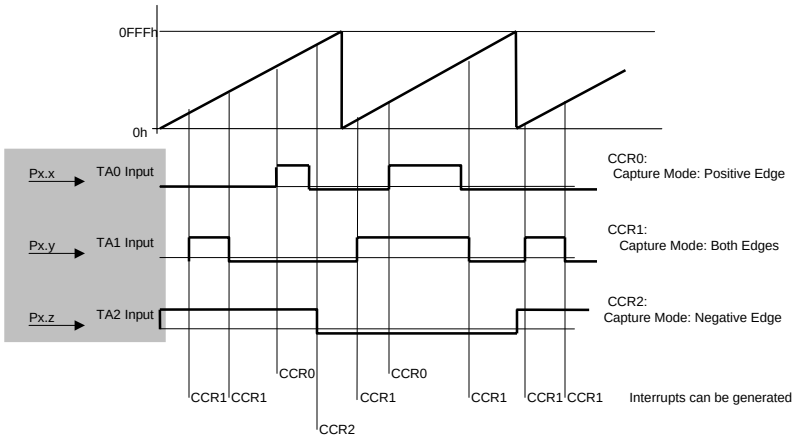


Timer_A Output Units



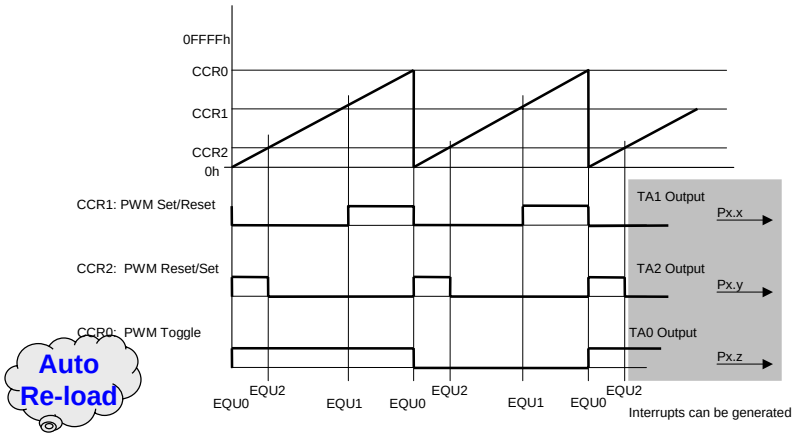
OMx2	OMx1	OMx0	Function	Operational Conditions
0	0	0	Output Mode	Outx signal is set according to Outx bit
0	0	1	Set	EQUx sets Outx signal clock synchronous with timer clock
0	1	0	PWM Toggle/Reset	EQUx toggles Outx signal, reset with EQU0, clock sync. with timer clock
0	1	1	PWM Set/Reset	EQUx sets Outx signal, reset with EQU0, clock synchronous with timer clock
1	0	0	Toggle	EQUx toggles Outx signal, clock synchronous with timer clock
1	0	1	Reset	EQUx resets Outx signal clock synchronous with timer clock
1	1	0	PWM Toggle/Reset	EQUx toggles Outx signal, set with EQU0, clock synchronous with timer clock
1	1	1	PWM Set/Reset	EQUx resets Outx signal, set with EQU0, clock synchronous with timer clock

Timer_A Continuous-Mode Example



Example shows three independent HW event captures.
CCRx "stamps" time of event - Continuous-Mode is ideal.

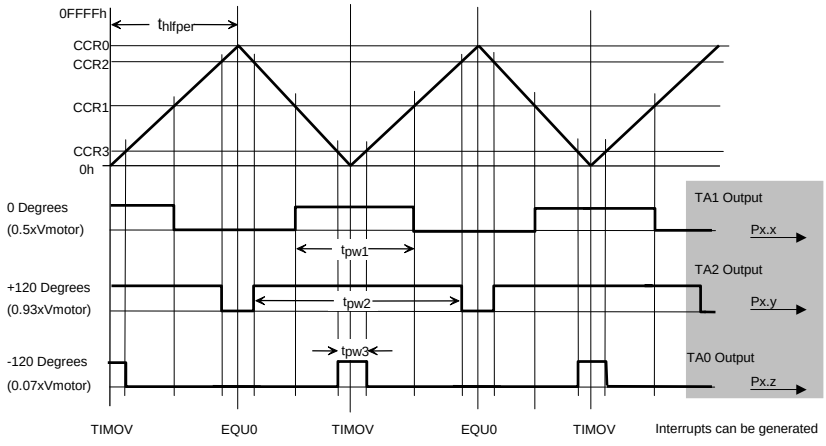
Timer_A PWM Up-Mode Example



Output Mode 4: PWM Toggle

Example shows three different asymmetric PWM-Timings generated with the Up-Mode

Timer_A PWM Up/Down Mode Example



Example shows Symmetric PWM Generation - Digital Motor Control

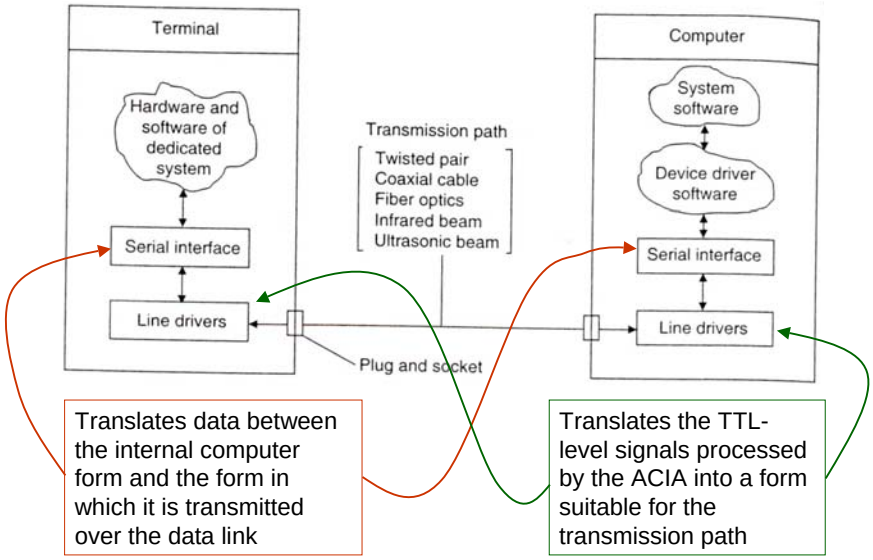
C Examples

```
//*******************************************************************
//  MSP-FET430P140 Demo - Timer_A Toggle P1.0,
//  CCR0 Contmode ISR, DCO SMCLK
//  Description; Toggle P1.0 using software and TA_0 ISR. Toggle rate is
//  set at 50000 DCO/SMCLK cycles. Default DCO frequency used for TACLK.
//  Durring the TA_0 ISR P0.1 is toggled and 50000 clock cycles are
      added to
//  CCR0.  TA_0 ISR is triggered exactly 50000 cycles. CPU is normally
      off and
//  used only durring TA_ISR.
//  ACLK = n/a, MCLK = SMCLK = TACLK = DCO~ 800k
//
//
//           MSP430F149
//         ---------------
//     /|\|             XIN|-
//      | |                |
//      --|RST         XOUT|-
//        |                |
//        |           P1.0|-->LED
//
//  M. Buccini
//  Texas Instruments, Inc
//  September 2003
//  Built with IAR Embedded Workbench Version: 1.26B
//  December 2003
//  Updated for IAR Embedded Workbench Version: 2.21B
//*******************************************************************
```

```
#include <msp430x14x.h>

void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;              // Stop WDT
  P1DIR |= 0x01;                         // P1.0 output
  CCTL0 = CCIE;          // CCR0 interrupt enabled
  CCR0 = 50000;
  TACTL = TASSEL_2 + MC_2; // SMCLK, contmode

  _BIS_SR(LPM0_bits + GIE); // Enter LPM0 w/ interrupt
}

// Timer A0 interrupt service routine
interrupt[TIMERA0_VECTOR] void TimerA(void)
{
  P1OUT ^= 0x01; // Toggle P1.0
  CCR0 += 50000; // Add Offset to CCR0
}
```


Serial Communication

Serial I/O Interface
Functional Units

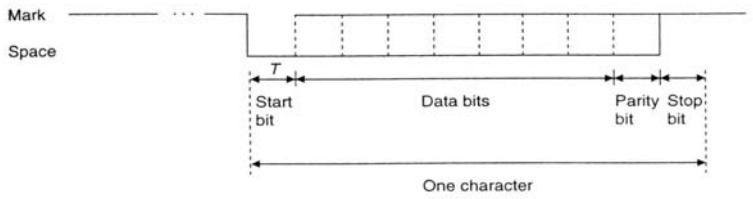


Asynchronous Serial Interface

- Asynchronous
  - Transmitted and received data are not synchronized over any extended period
  - No synchronization between receiver and transmitter clocks
- Serial
  - Usually character oriented
  - Data stream divided into individual bits at the transmitter side
  - Individual bits are grouped into characters at the receiving side
- Information is usually transmitted as ASCII-encoded characters
  - 7 or 8 bits of information plus control bits

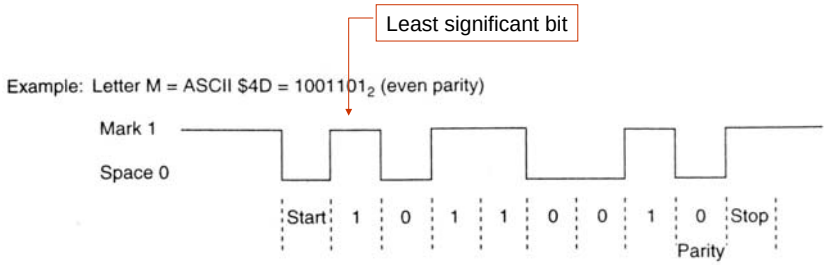
Asynchronous Serial Interface, cont'd

- MARK level (or OFF, or 1-state, or 1-level)
  - This is also the idle state (before the transfer begins)
- SPACE level (or ON, or 0-state, or 0-level)
- One character:
  - Start bit: space level
  - Data bits
  - Optional parity bit
  - Optional stop bit

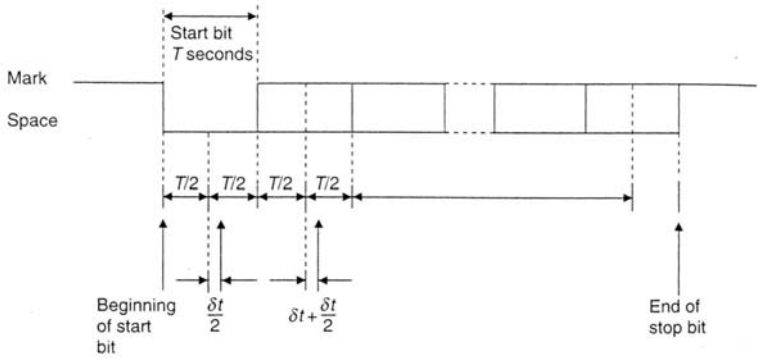


Asynchronous Serial Interface, cont'd

- 12 possible basic formats:
  - 7 or 8 bits of data
  - Odd, even, or no parity
  - 1 or 2 stop bits
  - Others exist also: no stop bits, 4/5/6 data bits, 1.5 stop bits, etc.



Receiver Clock Timing



- For N=9 bits (7 data + parity + stop) maximum tolerable error is 5% (assume that the receiver clock is slow -- [T + δt] instead of T)

  $T/2 > (2N+1)\delta t/2$

  $\delta t/2 < 1/(2N+1)$

  $\delta t/T < 100/(2N+1)$ as a percentage

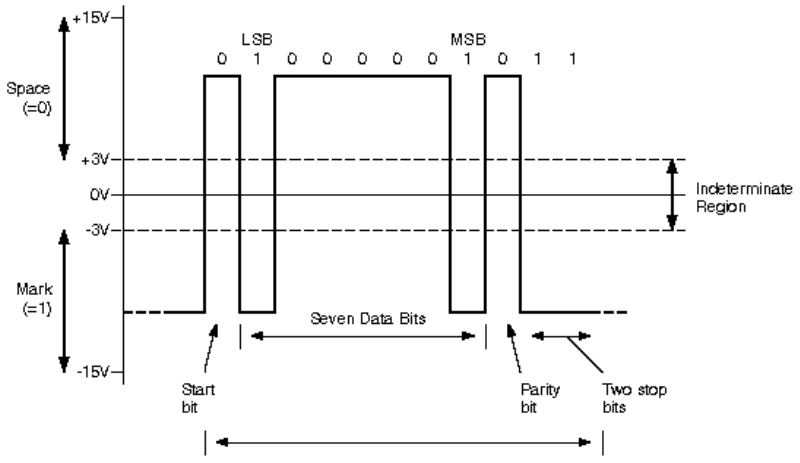
RS-232 Interface Standard

- Bi-polar:
  - +3 to +12V (ON, 0-state, or SPACE condition)
  - -3 to –12V (OFF, 1-state, or MARK condition)
- Modern computers accept 0V as MARK
- "Dead area" between –3V and 3V is designed to absorb line noise
- Originally developed as a standard for communication between computer equipment and modems
- From the point of view of this standard:
  - MODEM: *data communications equipment (DCE)*
  - Computer equipment: *data terminal equipment (DTE)*
- Therefore, RS-232C was intended for DTE-DCE links (not for DTE-DTE links, as it is frequently used now)

RS-232 Interface Standard

- Each manufacturer may choose to implement only a subset of functions defined by this standard
- Two widely used connectors: DB-9 and DB-25
- Three types of link
  - Simplex
  - Half-duplex
  - Full-duplex
- Basic control signals
  - RTS (Request to send):
    DTE indicates to the DCE that it wants to send data
  - CTS (Clear to send):
    DCE indicates that it is ready to receive data
  - DSR (Data set ready):
    indication from the DCE (i.e., the modem) that it is on
  - DTR (Data terminal ready):
    indication from the DTE that it is on

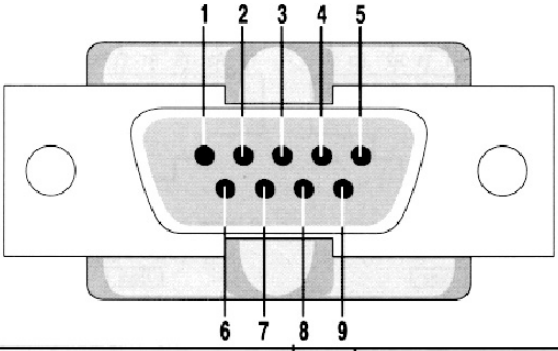
RS-232 Interface Standard, another example



- DTR (Data terminal ready): indication from the DTE that it is on

RS-232 Interface Standard

- DB-25 connector is described in the book; let's take a look at DB-9



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

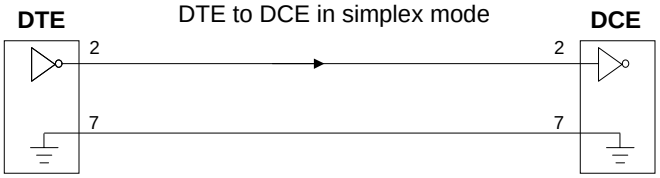
RS-232 Interface Standard

Example: 9 to 25 pin cable layout for asynchronous data

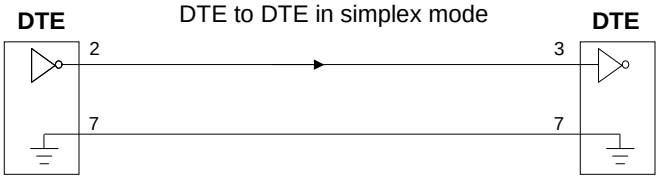
Description	Signal	9-pin DTE	25-pin DCE	Source DTE or DEC
Carrier Detect	CD	1	8	from Modem
Receive Data	RD	2	3	from Modem
Transmit Data	TD	3	2	from Terminal/Computer
Data Terminal Ready	DTR	4	20	from Terminal/Computer
Signal Ground	SG	5	7	from Modem
Data Set Ready	DSR	6	6	from Modem
Request to Send	RTS	7	4	from Terminal/Computer
Clear to Send	CTS	8	5	from Modem
Ring Indicator	RI	9	22	from Modem

The Minimal RS-232 Function

DTE to DCE in simplex mode

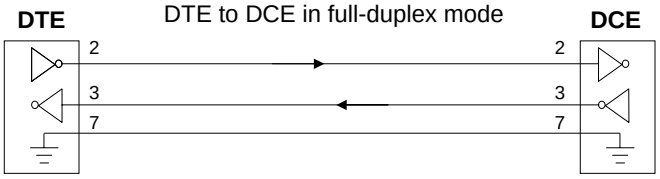


DTE to DTE in simplex mode

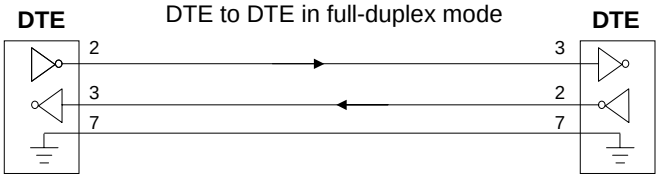


The Minimal RS-232 Function

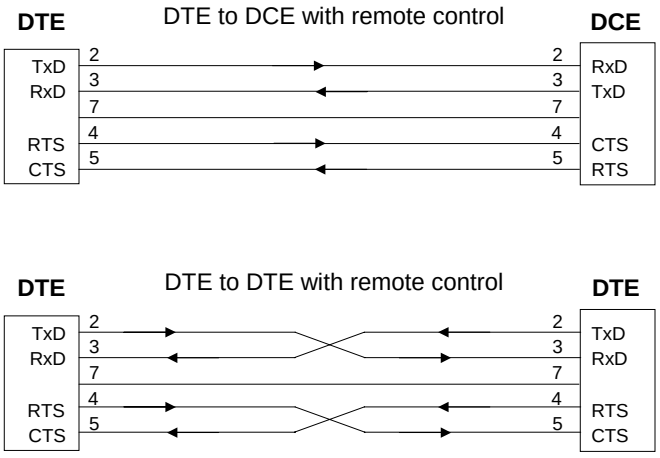
DTE to DCE in full-duplex mode



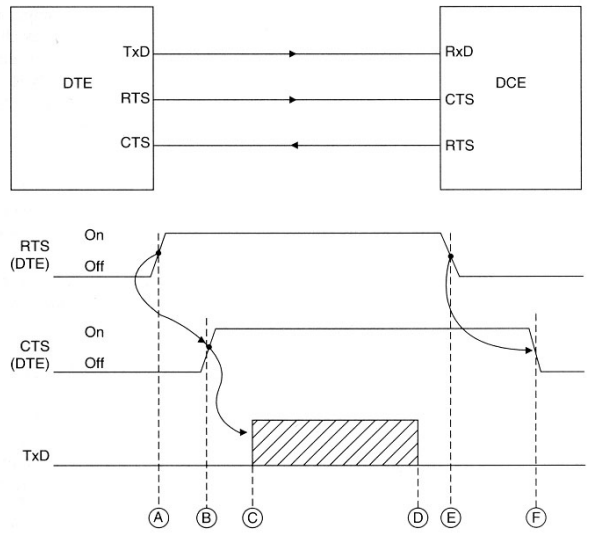
DTE to DTE in full-duplex mode



The Minimal RS-232 Function

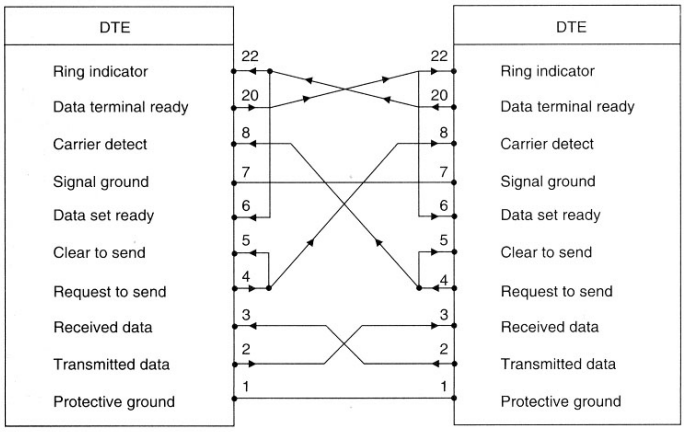


Handshaking Between RTS and CTS



Null Modem

- Null-modem simulates a DTE-DCE-DCE-DTE circuit



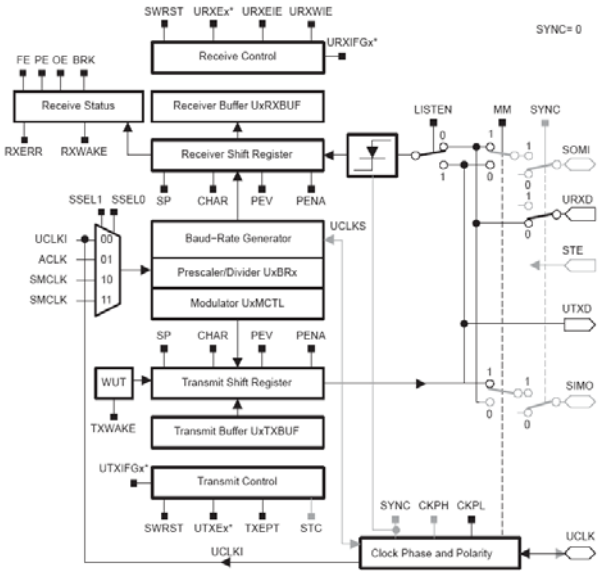
USART Peripheral Interface

- Universal Synchronous/Asynchronous Receive/Transmit (USART) peripheral interface supports two modes
  - Asynchronous UART mode (User manual, Ch. 13)
  - Synchronous Peripheral Interface, SPI mode (User manual, Ch. 14)
- UART mode:
  - Transmit/receive characters at a bit rate asynchronous to another device
  - Connects to an external system via two external pins URXD and UTXD (P3.4, P3.5)
  - Timing is based on selected baud rate (both transmit and receive use the same baud rate)

UART Features

- 7- or 8-bit data width; odd, even, or non-parity
- Independent transmit and receive shift reg.
- Separate transmit and receive buffer registers
- LSB-first data transmit and receive
- Built-in idle-line and address-bit communication protocols for multiprocessor systems
- Receiver start-edge detection for auto-wake up from LPMx modes
- Programmable baud rate with modulation for fractional baud rate support
- Status flags for error detection
- Independent interrupt capability for transmit and receive

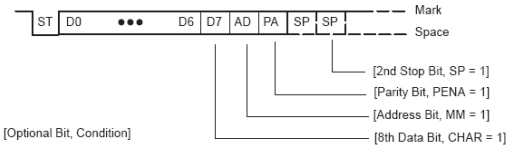
USART Block Diagram: UART mode



Initialization Sequence & Character Format

- Initialization Sequence
  - *Set SWRST bit*
  - *Initialize all USART registers with SWRST = 1*
  - *Enable USART module via the MEx SFRs (URXEx and/or UTXEx)*
  - *Clear SWRST via software (releases the USART for operation)*
  - *Optional: enable interrupts vie IEx SFRs*

- Character format



39